Proceedings of the International Conference on Automation, Computing and Renewable Systems (ICACRS 2022)
IEEE Xplore Part Number: CFP22CB5-ART: ISBN: 978-1-6654-6084-2

Machine Learning Techniques for Detecting DDoS Attacks in SDN

M. Kavitha
Department of CSE
Vel Tech Rangarajan Dr. Sagunthala R&D Institute of Science and Technology
Chennai, India
kavitha@veltech.edu.in

M.Suganthy
Department of ECE
Vel Tech Multi Tech Dr. Rangarajan Dr. Sakunthala Engineering College,
Chennai, Tamil Nadu, India Chennai, India
suganthym46@gmail.com

Aniket Biswas
Department of CSE
Vel Tech Rangarajan Dr. Sagunthala R&D Institute of Science and Technology
Chennai, India
vtu11309@veltech.edu.in

R.Srinivsan
Department of CSE
Vel Tech Rangarajan Dr. Sagunthala R&D Institute of Science and Technology
Chennai, India
srinivasanrajkumar28@gmail.com

R.Kavitha
Department of CSE
Vel Tech Rangarajan Dr. Sagunthala R&D Institute of Science and Technology
Chennai, India
rkavitha1984@gmail.com

A.Rathesh
Department of CSE
Vel Tech Rangarajan Dr. Sagunthala R&D Institute of Science and Technology
Chennai, India
vtu12500@veltech.edu.in

2022 International Conference on Automation, Computing and Renewable Systems (ICACRS) | 978-1-6654-6084-2/22/$31.00 ©2022 IEEE | DOI: 10.1109/ICACRS55517.2022.10029110

Abstract— Future internet is increasingly reliant on Software Defined Networking (SDN). With SDN, networks can be dynamically controlled, providing a global network. Compared to traditional networks, SDN offers the advantage of better security provisioning due to centralized management. However, SDN architecture manifests several new network security problems that need to be handled to improve the security of SDN networks. Information security and data analysis systems for Big Data have become more essential due to the increasing volume of data and its incremental growth. Monitoring and analyzing data is needed to detect any intrusion into a system or network via an intrusion detection system (IDS). By using traditional methods, traditional data analysis techniques are unable to detect attacks caused by high volumes, a wide variety and high speeds of network data. For an accurate and efficient data analysis process, IDS employs Big Data techniques. The paper uses machine learning models to detect Distributed Denial of Service (DDoS) attacks. The machine learning model is trained using data from KDD Cup 99.K Nearest Neighbor Classifier, Logistic Regression, and Decision Tree have been used to train and test the datasets. It can be concluded that machine learning methods can be more effective at detecting DDoS attacks than traditional methods, that can be applied to software defined networks. Several experiments demonstrate the potential of our proposal to detect intrusion in SDN environments after extensive evaluation.

Keywords—Software Defined Networks, Decision Tree, Machine Learning, Distributed Denial of Service, Intrusion Detection System

I. INTRODUCTION

Over the past three decades, the Internet architecture has evolved from a simple network to a complex system. Decentralized, autonomous, and consisting of a variety of network devices (with complex protocols, the system is largely decentralized and autonomous. These components are usually developed by manufacturers for networks. Manufacturers have to develop their own designs, firmware, and software in order to operate their own hardware. In light of the changing nature of applications and the endlessly changing demands for data, the existing Internet architecture is unsuitable for adapting to them. SDN [1] has been introduced for configuring and deploying networks that offers unparalleled flexibility. Managing traffic more effectively can be improved by separating the control plane and data plane. SDN architectures, however, pose a whole host of security challenges. Recent concerns regarding SDN security have gained considerable attention, as shown in [2] and [3]. A network intrusion detection system (IDS) is a key security tool. IDSs based on signatures identify new attacks using a database of signatures from previous attacks. Patterns in an anomaly database are used to identify attacks in an anomaly-based IDS. Zero-day attacks will not be detected by the system, but false alarms are rare. A signature-based IDS is highly effective if it has a current signature database with accurate data. Maintaining databases of this type in real time is difficult and incurs high operational expenses. Analyzing observations that differ from the baseline allows one to identify anomalies using anomaly-based IDS. An IDS based on signatures is less likely to detect zero-day attacks. In order to find anomalies on the network, the process monitors real-time traffic on the network. Networks are growing in size, big data is reaching an unprecedented volume, and powerful computation facilities are making all processes more complex and real-time. In order to manage network-based intrusions effectively, these systems must be analyzed carefully, precisely, and accurately, which has not been possible in the past. On the other hand, machines have improved their accuracy tremendously with AI algorithms. A growing need for better performance is leading to its introduction among the network types. In the modern era of modern cyber security risks, network-based intrusion detection systems

(NIDS) have become a more realistic possibility due to their implementation via software defined networks (SDNs). Rapid growth of network data and devices pose a high risk of security threats. By deploying machine learning algorithms over SDN, the paper implements an Intrusion Detection System. KNN Classifier, Decision Tree Classifier, and Logistic Regression algorithms are used to analyse the data.

II. Related Works

Algorithms for machine learning that integrate SDN have attracted considerable interest. An experimental study was undertaken in [26] to find a solution that fixed problems in KDD Cup 99 using the NSL-KDD dataset for intrusion detection. The five Machine Learning algorithms were studied. NSL-KDD dataset contained only 13 features due to correlation feature selection algorithm, resulting in a reduced complexity. The NSL-KDD dataset is used to detect network anomalies in real-world networks. The average accuracy is achieved for all 41 features using these classic machine learning models. After reducing the number of features to 13, the same models were trained again and scored an average accuracy of 98%, 85%, 95%, 86%, and 73%. Based on deep neural networks, a system for intrusion detection and finding was proposed in [4]. In the proposed method, six basic and traffic characteristics are derived from the NSL-KDD dataset, which is easily derived from the SDN infrastructure. Accuracy, precision, and recall are combined to yield a F1-score of 0.75. For the second evaluation, [5] implemented seven classic machine learning models. A neural network has been applied to detect SDN anomalies with up to 89% accuracy [6]. Feature scaling is also improved and accelerated via the Min-Max normalization technique. An intrusion detection application used SVM classifiers coupled with principal component analysis (PCA). A model for detecting abnormal patterns is trained and optimized in this approach. It was proposed to use Min-Max normalization to reduce the error rates [7]. Radial basis function kernels were used for optimization. In total, an average accuracy of 95% was achieved using the 31 features of the dataset, as well as the metrics used to evaluate the proposed model. In [8], XGBoost was used to differentiate between DoS attacks and non-DoS attacks. A prototype and development technique based on SDN using POX SDN was evaluated and analyzed using Future Internet 2021, 13, 111 5 of 18. The network topology was simulated with Mininet to simulate cloud detection based on SDN in real-time. A logarithmic-based and a Min-Max-based normalization was applied. In comparison to RF and SVM, XGBoost had an overall accuracy of 98%, 96%, and 97% [9]. Based on the packet network, a six-tuple classification of characteristics is proposed. Speeds of flow packets (SSIP), speeds of source ports, deviations of flows bytes (SDFB), speeds of flow entries, and ratios of flow entries and pairs should be investigated. SVM based on the six characteristics calculates the current state of the network to be either normal or attack. For an average accuracy of 95%, attack flow, defense flow, and defense array were selected [9]. In order to detect attacks in a multi-class classification, TSDL was developed and implemented. The detection rate and monitoring efficiency were improved using down-sampling and other preprocessing techniques applied to various datasets. 89.134% of UNSW-NB15 samples were correctly identified. There have been several neural network models proposed for NIDS and various datasets were used to implement the proposed approach to distinguish between normal and attack packets in the network. In general, factors like those are intended to provide neural networks with the capability of learning complex patterns with multiple scopes of information within the same packet, but not always to do that. This model is based on four hidden layers and used to monitor intrusion attacks on KDD cup99[8]. Data preprocessing was performed by using feature scaling and encoding. Various datasets were analyzed using more than 50 features. Due to the large number of features, complex GPUs were required to reduce training time. It was proposed that NIDS use a supervised adversarial auto-encoder neural network [9][10]. The generator and discriminator in GANS are different neural networks in contention with each other. With the Jensen-Shannon minimization algorithm, the objective function will be minimized as much as possible. By generating fake data packets, the generator attempts to generate an attack, but the discriminator determines whether these packets are valid or not; in other words, it determines whether these packets are real or fake [11]. A regularization penalty is also applied to the model structure for overfitting control behavior. U2RL and R2L detected reasonably well, but other tests revealed lower detection rates. The paper [12][13] presented multichannel deep learning of features for NIDS based on CNNs, two fully connected layers and a SoftMax classifier. With an average accuracy of 94%, the evaluation is conducted over different datasets. However, the attack structure and characteristics were not clearly outlined in the proposed model. The attribute-based encryption is discussed in [14][15].

III. Methodology

To train the model, data is analyzed, best features are extracted, and preprocessing techniques are performed using only best five features, based on the best hyperparameters. The dataset consists of 2827876 rows and 80 columns, which is collected from KDD Cup99 database. The normalization helps to process the raw data and classifies datasets which can be used in analysis using machine learning algorithms. It helps in identifying the attacks and allows to classify the raw data into different categories. To classify attacks, in which machine learning algorithms are used. It is then analyzed to determine which type of attack is involved and what action is required. As a result, the processed data are placed in the algorithm and classified as normal or an attack. Fig 1 illustrates the methodology used; data is processed with KNN, Logistic Regression and Decision Tree Classifier, with Fig 3 showing that Decision Tree Classifier classifies better with a higher accuracy rate.

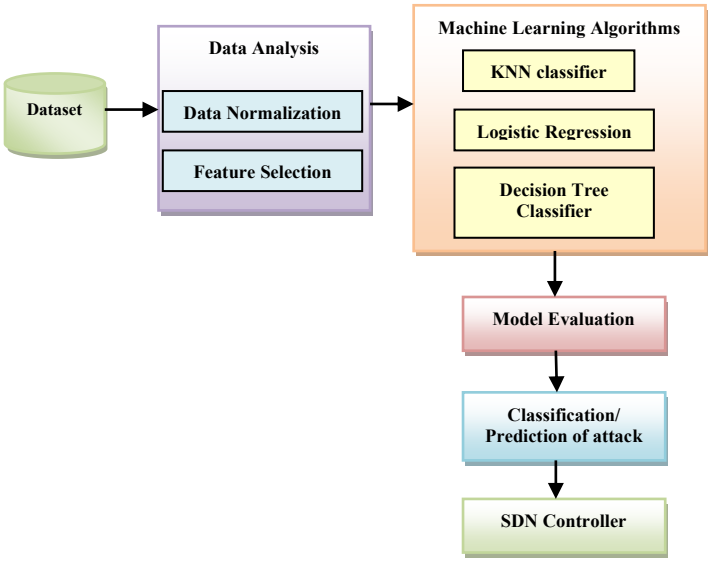

Fig. 1. Proposed Architecture to detect DDoS Attack

There are various attacks included in Table I of the data set and it also desrcibes about the data used. In total, there are 1113112 total cases, 46.5% of which are benign and 53.5% of which are malignant. 80% of the data come from the training set while 20% comes from the testing set.

TABLE I. Attacks occurs in the Network

Attacks	Number of features
BENIGN	2271320
DoS Hulk	230124
PortScan	158804
DDoS	128025
DoS Golden Eye	10293
FTP-Patator	7935
SSH-Patator	5897
DoS slowloris	5796
DoS Slowhttptest	5499
Bot	1956
Web Attack - Brute Force	1507
Web Attack - XSS	652
Infiltration	36
Web Attack - Sql Injection	21
Heartbleed	11

IV. Experimental Results

A. *Algorithm 1 - K Nearest Neighbor classifier*

Input : Training Data

1. There are K neighbors selected
2. Computing Euclidean distance

$$ED=\sqrt{(X2-X1)^2+(Y2-Y1)^2}$$

3. The Euclidean Distance is calculated using K nearest neighbors
4. K nearest neighbors are counted within each category
5. A new point of data is added to the category with the most neighbors.
6. Test and evaluate the model

- Data preprocessing process
- Implementing the K-NN algorithm on the training data
- Prediction of test results
- Confusion matrix creation (test accuracy of result)
- Visualizing results from a test set
-

A training set is D, a testing set is Z, and a nearest neighbour number is K. Samples are represented by (x1, x2, x3, x4, x5, ..., xn, y), where x1-xn correspond to n attributes and y corresponds to their label. There are many advantages of KNN algorithm, including its simplicity, convenience, and ease of use. No parameters need to be estimated, no threshold needs to be considered, and no training is necessary. Classifying test samples, however, results in a large amount of computation as well as a large amount of memory and CPU usage. Furthermore, it is essential to select K values that are appropriate if the samples are unbalanced. It is also possible that too much data will result in too much weight in the results if the range of data is too broad. To Make normal data collection and attack data collection timed at 1:1 in order to overcome the problem of unbalanced sample data. To ensure a suitable K value is used for subsequent classifications of unknown flow samples, the KNN algorithm model will be used for classifying and predicting samples after obtaining the dataset (training set). Using the same dataset, any subsequent classifications of unknown flow samples will be performed after the algorithm has tested multiple K values.

Algorithm 2 - Logistic Regression

Input: Dataset(Training data)

1. For i=1 to n
2. For each instance of data ti
3. Regression targets should be set to

$$\mathbf{Z_i = y_i - P\ (1|a_j)\ /\ [P\ (1|a_j).\ (1\text{-}P\ (1|a_j))]}$$

4. Initiate the weight for instance aj to be

$$P(1*a_j).(1\text{-}P(1*a_j)$$

5. Create a f(j) with weights (Wj) and values as class

(Zj)

6. Class is 1, if P (1|dj) > 0.5, else class is 2

The first step is to classify inputs as zero or one. The first step in determining the class of a training set is to calculate the probability (or class score) of the set being in class 1. According to the below function, the y parameter is .

P(y=1)= 1 1+e −(β 0+ β 1 x 1+...+β n x n)

A high probability is predicted for observations belonging to class 1 and a low probability for observations belonging to class 0 by selecting coefficients that maximize the likelihood of predicting those observations. For Classifying input values into classes by defining a threshold boundary, the next step is to define a threshold boundary for the classifier. In general, a threshold value of around 0.5 is chosen based on our business problem. In other words, if the probability is larger than 0.5, such observations are classified as class 1 types, while the rest are classified as class 0 types.

It depends on the type of error, which can be either false positives or false negatives, when choosing a threshold value. Observations that belong to class 0 are mistakenly predicted as class 1 by the model, which results in a false positive error. Models predict class 0, but observations belong to class 1. Hence, a false negative error occurs. It would be perfect if all the 1s and 0s (or true and false) were classified correctly.

C. Algorithm 3 - Decision tree

Input: Training data

1. Construct node M
2. In case sample S and sample N belong to the same Class C, N will be returned as a leaf node
3. End if
4. Leaf node is returned, M as a class with the higher part
   S if A = Φ or the attribute values are the same
5. End if
6. Calculate the Gain ratio to find the splitting Attribute

$$GainRatio(S,A) = \frac{Gain(S,A)}{SplitInformation(S,A)}$$

7. Build a tree

Output : Decision tree

In Decision-Tree (DT) algorithms, classification rules are contained in the internal nodes of the tree, whereas class labels are contained in the leaf nodes. This is an active learning algorithm based on predictive modelling. DTs predict class labels for test samples based on the rules derived from the supplied features during the training stage. DDoS attacks are differentiated from benign flows using these classification rules. Figure 2 shows the accuracy of the Classification

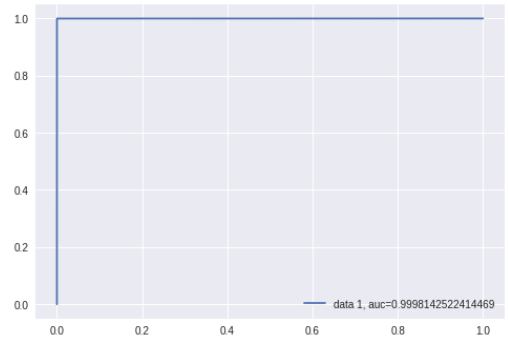

Fig 2. Accuracy of Decision Tree Classifier

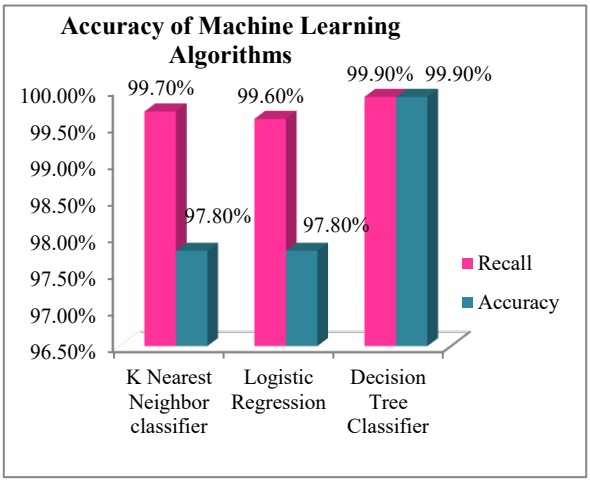

Fig 3 : Accuracy and Recall of Algorithms Used

Python is used to analysis the dataset and predict the results. The accuracy and recall is calculated using the below formulas.

Accuracy = True Negative + True Positive / True Negative + False Positive+True Positive + False Negative.

Recall = + True Positive / True Positive + False Negative

V. CONCLUSION

In the last two decades, machine learning algorithms in SDN is paying more attention due to their ability to use data and algorithms for better security detection and efficiency. Training and testing were performed using KDD cup99, a benchmarking dataset. The algorithm's performance is improved and optimized through feature selection, feature normalization, and data preprocessing techniques, allowing for an efficient and successful training process. The objective is to select the optimal algorithm by comparing the algorithms; KNN, Logistics Regression, and Decision Trees. Using a metric such as recall and accuracy,

we analyze the advantages and disadvantages of implementing any one or more of them. The Decision Tree model outperformed the other algorithms based on the evaluation metrics. Using the proposed method, real-time attacks can be detected and protected against on the SDN platform. Moreover, future research will examine more metrics of evaluation. The approach is expected to be implemented using Deep Learning algorithms.

REFERENCES

[1] Dey, S. K., & Rahman, M. M. (2019). Effects of machine learning approach in flow-based anomaly detection on software-defined networking. Symmetry, 12(1), 7.

[2] Gao, M., Ma, L., Liu, H., Zhang, Z., Ning, Z., & Xu, J. (2020). Malicious network traffic detection based on deep neural networks and association analysis. Sensors, 20(5), 1452.

[3] Nobakht, M., Sivaraman, V., & Boreli, R. (2016). A host-based intrusion detection and mitigation framework for smart home IoT using OpenFlow. In 2016 11th International conference on availability, reliability and security (ARES) (pp. 147-156). IEEE.

[4] Sebbar, A., Zkik, K., Baddi, Y., Boulmalf, M., & Kettani, M. D. E. C. E. (2020). MitM detection and defense mechanism CBNA-RF based on machine learning for large-scale SDN context. Journal of Ambient Intelligence and Humanized Computing, 11(12), 5875-5894.

[5] Ngo, D. M., Pham-Quoc, C., & Thinh, T. N. (2020). Heterogeneous hardware-based network intrusion detection system with multiple approaches for SDN. Mobile Networks and Applications, 25(3), 1178-1192..

[6] Bag, S., Gupta, S., & Wood, L. (2020). Big data analytics in sustainable humanitarian supply chain: Barriers and their interactions. Annals of Operations Research, 1-40.

[7] Lazarevic, A., Kumar, V., & Srivastava, J. (2005). Intrusion detection: A survey. In Managing cyber threats (pp. 19-78). Springer, Boston, MA.

[8] Sultana, N.; Chilamkurti, N.; Peng, W.; Alhadad, R. Survey on SDN based network intrusion detection system using machine learning approaches. Peer Netw. Appl. 2019, 12, 493–501.

[9] Chen, P. J., & Chen, Y. W. (2015, September). Implementation of SDN based network intrusion detection and prevention system. In 2015 International Carnahan Conference on Security Technology (ICCST) (pp. 141-146). IEEE.

[10] Amrish, R., K. Bavapriyan, V. Gopinaath, A. Jawahar, and C. Vinoth Kumar. "DDoS Detection using Machine Learning Techniques." Journal of IoT in Social, Mobile, Analytics, and Cloud 4, no. 1 (2022): 24-32.

[11] Mugunthan, S. R. "Novel Cluster Rotating and Routing Strategy for software defined Wireless Sensor Networks." Journal of ISMAC 2, no. 02 (2020): 140-146.

[12] Li, Y., & Lu, Y. (2019, September). LSTM-BA: DDoS detection approach combining LSTM and Bayes. In 2019 Seventh International Conference on Advanced Cloud and Big Data (CBD) (pp. 180-185). IEEE.

[13] Ring, W. (2019). Hotho, 2019 Ring M., Wunderlich S., Scheuring D., Landes D., Hotho A. A survey of network-based intrusion detection data sets, Comput. Secur, 86, 147-167.

[14] Ahmad, R., & Alsmadi, I. (2021). Machine learning approaches to IoT security: A systematic literature review. Internet of Things, 14, 100365.

[15] Aleesa, A. M., Zaidan, B. B., Zaidan, A. A., & Sahar, N. M. (2020). Review of intrusion detection systems based on deep learning techniques: coherent taxonomy, challenges, motivations, recommendations, substantial analysis and future directions. Neural Computing and Applications, 32(14), 9827-9858.